REMARKS

Independent Claims 1, 22, and 26 are amended in a further effort to define patentable subject matter over the applicable art of record. Claims 2, 21, 23, 24, 28, and 29 are cancelled without prejudice. Claims 1, 3-20, 22, 25, 26, and 30 remain, with no claim previously allowed.

Claims 1-6, 8-26, and 28-30 were rejected as unpatentable over *Nelson* (US 6,418,346) in view of newly-cited *Smith* (US 2005/0065678). The Applicant respectfully traverses that rejection, as possibly applied to the amended claims.

Claim 1 is amended to define a portable WAP-enabled diagnostic device for troubleshooting a WAP network having a plurality of elements. The claimed diagnostic device includes a microbrowser module configured for communicating with an external source of information including a WAP network undergoing troubleshooting. The diagnostic device further comprises a diagnostic module configured for analyzing information associated with elements of the WAP network in a predetermined sequence. That predetermined sequences configured to emulate a process flow of signals through the elements of the WAP network undergoing diagnoses. A display screen is coupled to the processor of the diagnostic device and displays information pertaining to operating perimeters of the WAP network undergoing diagnoses. By analyzing information associated with each element of the WAP network in a sequence that emulates the process flow of signals through elements of the WAP network, the diagnostic device according to Claim 1 allows a user to sequentially eliminates certain elements of the WAP network as the source of difficulty during troubleshooting, and may possibly

identify this specific source of the difficulty. (Support for the above-discussed revisions to Claim 1 is found in the specification, for example, at paragraph 0031.)

The rejection asserts that *Nelson* discloses the elements of Claim 1 (prior to the current amendments) except for teaching a diagnostic module configured for analyzing information received from an external source in a predetermined sequence configured to emulate a process for diagnoses. *Smith* is cited as supplying the teaching missing from *Nelson*. The Applicant traverses the Examiner's analysis of *Nelson* and the teachings of *Smith* available against the present application.

Nelson discloses a system for transferring data into and out of medical devices. In that reference, data from a source such as an implanted pace maker is transferred first to a personal data monitor (PDM) carried by the patient. That data is then transferred by telemetry to a programming device 20, which may be at a remote location via a communication network. Although *Nelson* mentions "diagnosis", that reference does not disclose or teach a diagnostic module or any other element that is configured for analyzing information, much less for analyzing information associated with elements of a WAP network, or information in a predetermined sequence configured to emulate a flow of signals through elements of a WAP network. To the contrary, and keeping with the traditional physician-patient relation in medical treatment, *Nelson* teaches that <u>diagnosis</u> is accomplished by a physician or other caregiver, relying on information communicated from the patient using *Nelson*'s system (column 12, lines 13-37).

Turning to *Smith*, that application was filed May 24, 2004 and, as such, is not available as a reference against the present application having a filing data of February 14, 2002.

However, *Smith* claims priority on several related US applications. Of those related applications, only Application Nos. 09/640,785 filed August 18, 2000; provisional Application No. 60/351,165 filed January 23, 2002; and provisional Application No. 60/354,673 filed February 5, 2002 predate the filing of the present application. That is, the cited *Smith* published application (2005/0065678) is available as a reference against the present application only for disclosure finding adequate support in one of the prior Applications 09/640,785; 60/351,165; or 60/354,673.

The undersigned has obtained copies of those three priority applications, which are available on PAIR. A courtesy copy of each such document is submitted with this response, for the convenience of the Examiner. As the following discussion shows, the disclosure material for which *Smith* is cited lacks support in any of those three priority applications.

The rejection cites *Smith* as disclosing an enterprise resource planning system with integrated vehicle diagnostic and information system including a diagnostic module for analyzing information received from the external source of information. Paragraphs 0024 and 0030-0031 are cited in support. Paragraphs 0025 and 0032 of *Smith* are also cited as disclosing the diagnostic module analyzing information in a predetermined sequence configured to emulate a process for diagnoses. However, neither the cited paragraphs nor the substantive disclosures within those paragraphs appear in any of the three priority applications that predate the filing of the present application. Priority Application 09/640,785 discusses (page 4, lines 17-25) a method of remotely reading at least one vehicle parameter from a selected vehicle in a fleet, changing that parameter on command sent from the remote location, and receiving an acknowledgement of the

command from the vehicle. Page 12, lines 16-20 of that '785 application discuss vehicle onboard units having access to a plurality of controllers or discrete measurement points for accessing data within the vehicle. Page 20, lines 24-29 of that application discuss sending reprogramming commands to specific vehicles and setting pre-defined time periods wherein parameter readings should be taken for specific vehicles within a fleet. However, not in those passages or elsewhere does the '785 application disclose a diagnostic module configured for analyzing information associated with elements of a WAP network in a predetermined sequence configured to emulate a process flow of signals through the elements of that network.

The two provisional Applications 60/354,673 and 60/351,155 do not supply the aforementioned teaching deficiencies of the '785 application. Provisional Application '673 merely discusses a standard infrastructure for establishing data communication between a vehicle and a remote location. Provisional Application '165 discusses architecture of a wireless communication system for creating communication-based applications, again in the context of communications between a mobile device and a host server. Neither provisional application mentions a diagnostic module for receiving and analyzing a flow of signals, through the elements of a WAP network or elsewhere in a predetermined sequence configured to emulate a process flow of signals. Those provisional applications, as well as the '785 nonprovisional application, are concerned with storing and processing vehicle information from a remote location, and dealing with the information itself (brakes, engine, transmission information), not with diagnosing the operation of a network itself. Accordingly, to the limited extent that the applicable priority support for *Smith* contains information available against the present application,
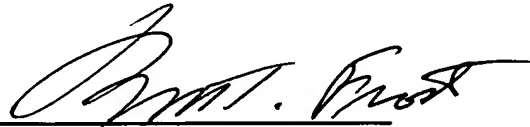
11

that information combined with *Nelson* would not have placed one of ordinary skill in the relevant art, in possession of the invention defined in Claim 1 et al. Those claims are, accordingly, patentable over the applied art.

Independent Claims 22 and 26 are amended to contain limitations making the combinations of those claims specific to analysis and diagnosis of a WAP network. The foregoing arguments regarding patentability over *Nelson* and *Smith* (to the limited extent that document is available as a reference) apply as well to Claims 22 and 26, and to the claims depending therefrom.

The foregoing is submitted as a complete response to the Office action identified above. The Applicant respectfully submits that all claims remaining in this application are patentable over the applied art and solicits a notice to that effect.

Respectfully submitted,

MERCHANT & GOULD

Date: August 4, 2006

Roger T. Frost
Reg. No. 22,176

Merchant & Gould, LLC
P.O. Box 2903
Minneapolis, MN 55402-0903
Telephone: 404.954.5100

39262
PATENT TRADEMARK OFFICE

SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR REMOTE VEHICLE DIAGNOSTICS, MONITORING, CONFIGURING AND REPROGRAMMING

Inventors: William Bromley
                 Brian R. Carl
                 Sam Chang
                 Brian Crull
                 Andrew Ditchfield
                 Dennis Essenmacher
                 Jerry Hosak
                 Michael Kapolka
                 Sam Konyn
                 Thomas Nagi
                 Charles Wright

Background of the Invention

Field of the Invention

The present invention relates generally to computer data and information systems, and more particularly to computer tools for storing, processing, and displaying fleet vehicle information.

Related Art

In today's business environment, it is common for companies to own a large amount (i.e., a fleet) of motor vehicles. A company, depending on their particular line of business, may have a fleet of passenger cars, light trucks, vans, heavy trucks or any combination of theses types of vehicles. Typical examples of such companies include commercial courier services, moving companies, freight and trucking companies, as well as passenger vehicle leasing companies and passenger carriers.

Such companies must typically manage each of the hundreds of vehicle within their fleets. The most critical management operations include the maintenance and repair, and maximizing the efficiency of these vehicles. In addition, timely reporting of key information related to the vehicle, such as

mileage, trip information, fluid status, and other parameters must be available in a timely fashion. In order to maximize profits, a company must maximize the amount of time each vehicle spends performing its intended function. That is, a company must minimize the amount of time each vehicle spends in a service environment (i.e., a repair and maintenance facility). Further complicating the situation is the fact that the vehicles within a company's fleet may operate throughout the nation's roads, but repair and maintenance facilities and vehicle configuration facilities are sparsely located in certain geographic locations.

One management technique has traditionally been to schedule vehicles for routine inspections on a rotating basis. While this technique has improved efficiency somewhat, it still involves taking a percentage of the fleet's vehicles out of service when in fact, they may not need to be in a service environment or may not be available to be serviced or configured.

One development has led to the decrease in the amount of time vehicles needed to be in the service environment during routine inspections. That is, during the '70s and early 1980's manufacturers started using electronic means to control engine functions and diagnose engine problems. This effort was primarily motivated to meet new and tougher Environmental Protection Agency (EPA) emission standards. Nevertheless, onboard diagnostic systems eventually became more sophisticated. Vehicles today typically include several controllers attached to a vehicle data bus that allow the engine and parts of the vehicle's chassis, body and accessory devices to be monitored.

Several instruments were designed to take advantage of vehicles onboard diagnostic and control systems. First, there were large pieces of equipment to perform diagnostics and these were followed by hand-held devices. These instruments increased the speed and efficiency of vehicle maintenance and configuration. Such instruments, however, did not eliminate the need for vehicles, which may be operating nation-wide, to be brought to a centralized (or regional) repair and maintenance facility. That is, these devices needed to be connected directly to the vehicle. Further, there still has not been any systematic way for companies to remotely diagnose, monitor or configure their fleet's

vehicles. That is, routine maintenance or configuration on a rotating basis is arbitrary and not based on which specific vehicles really require service.

Therefore, given the above, what is needed is a system, method, and computer program product for remote vehicle diagnostics, monitoring, configuring and reprogramming. The system, method, and computer program product should allow fleet managers, without heavy infrastructure additions, to take advantage of today's vehicle's onboard diagnostic systems, computer advances, and mobile communications in order to remotely diagnose, monitor and reprogram their fleet's vehicles.

Summary of the Invention

The present invention meets the above-mentioned needs by providing a system, method, and computer program product for remote vehicle diagnostics, monitoring, configuring and reprogramming.

The system of the present invention allows a user to perform total fleet logistics by facilitating vehicle parameter changes, vehicle health tracking, and receipt of vehicle maintenance need indications, thus eliminating the need to physically bring vehicles to a repair and maintenance facility. More specifically, the system includes a plurality of vehicles each having an onboard unit as described herein. The onboard unit is coupled to the vehicle data bus of each of the plurality of vehicles, which in turn is connected to the vehicle's several controllers.

The system further includes an application server which provides the user with a graphical user interface (GUI) (e.g., Web pages over the Internet) in order to send and receive data from each of the plurality of vehicles. A repository database, accessible via the application server, is also included which stores information related to the subscribers of the system and the specifics in relation to the vehicles in their fleet.

An onboard unit server, coupled to the application server, is also included which contains means to convert command data between a format understandable

by the user using the GUI (e.g., change max cruise speed to 55 MPH") and a format understandable by the vehicle data bus of each of the plurality of vehicles (e.g., a binary data stream). Finally, the system includes a communications means, coupled to the onboard unit server, for handling (mobile) communications between the onboard unit server and the onboard units located on each of the plurality of vehicles.

The method and computer program product of the present invention includes the steps of accessing the repository database in order to provide the user with a list of specific vehicles within the fleet and the vehicles' associated vehicle parameters. Next, a command from the user is received via the GUI. The command typically includes information specifying at least one vehicle within the fleet and at least one vehicle parameter. Then, the command is stored in the repository database along with the time and date that the command was received from the user. Next, the command is converted from a format understandable by the user using the GUI, to a format understandable by the vehicle data bus of the at least one vehicle within the fleet.

The method and computer program product of the present invention further includes sending the command, via a wireless mobile communications system to the onboard unit located on the targeted vehicle within the fleet. This causes the previously specified vehicle parameter to be read or changed (depending on whether, for example, the command was related to diagnostic or reprogramming activities respectively). Next, an acknowledgment of the command is received from the vehicle via the wireless mobile communications system. Finally, the acknowledgment is stored in the repository database so that the user may later retrieve it using the GUI.

One advantage of the present invention is that it allows a large fleet (e.g., several hundred) of commercial vehicles (e.g., a fleet of commercial delivery vans and/or trucks), of different makes and models, to be remotely configured, monitored, re-calibrated, and diagnosed without having to be brought to a centralized location (e.g., company headquarters). That is, the present invention provides a means for obtaining "total population" vehicle information.

Another advantage of the present invention is that it provides tampering alert notification should any vehicle parameter be changed without authorization once the vehicle leaves a company location or headquarters.

Another advantage of the present invention is that it provides users (e.g., fleet managers, vehicle distributors, vehicle dealers and the like) with a consistent graphical user interface, regardless of the vehicle makes and models that comprise their fleet.

Another advantage of the present invention is that it enables users to obtain real-time fleet characteristics, trend analysis and diagnostics, as well as allow fleet managers to provide real-time driver/fleet notification.

Yet another advantage of the present invention is that it allows parametric data capture, diagnostic code capture, trip data capture, system reconfiguration, system re-calibration, and correlation analysis to be performed on a fleet of vehicles on a customer-specified schedule.

Further features and advantages of the invention as well as the structure and operation of various embodiments of the present invention are described in detail below with reference to the accompanying drawings.

Brief Description of the Figures

The features and advantages of the present invention will become more apparent from the detailed description set forth below when taken in conjunction with the drawings in which like reference numbers indicate identical or functionally similar elements. Additionally, the left-most digit of a reference number identifies the drawing in which the reference number first appears.

FIG. 1 is a block diagram illustrating the system architecture of an embodiment of the present invention, showing connectivity among the various components;

FIG. 2A is a block diagram of the physical architecture of an onboard unit according to a preferred embodiment of the present invention;

FIG. 2B is a block diagram of the software architecture of an onboard unit according to a preferred embodiment of the present invention;

FIG. 3 is a flowchart depicting an embodiment of the operation and control flow of the remote vehicle diagnostics, monitoring and reprogramming tool of the present invention;

FIGS. 4A-4B are windows or screen shots, relating to vehicle alerts, generated by the graphical user interface of the present invention;

FIGS. 5A-5C are windows or screen shots, relating to vehicle parameter readings, generated by the graphical user interface of the present invention;

FIGS. 6A-6B are windows or screen shots, relating to vehicle parameter reprogramming, generated by the graphical user interface of the present invention; and

FIG. 7 is a block diagram of an exemplary computer system useful for implementing the present invention.

Detailed Description of the Preferred Embodiments

TABLE OF CONTENTS

I.   *Overview*

The present invention relates to a system, method, and computer program product for remote commercial vehicle diagnostics, monitoring, configuring and reprogramming. The remote vehicle diagnostics, monitoring, configuration and reprogramming tool described herein will become essential to any business concern which deals with commercial fleet maintenance and service operations (i.e., it is a "total fleet logistics" tool).

In an embodiment of the present invention, an application service provider provides and allows access, on a subscriber basis, to a remote vehicle diagnostics, monitoring, configuration and reprogramming tool via the global Internet. That is, the application service provider would provide the hardware (e.g., servers) and software (e.g., database) infrastructure, application software, customer support, and billing mechanism to allow its customers (e.g., fleet managers, vehicle distributors, vehicle dealers, original equipment manufacturers (OEM), leasing/rental companies, and the like) to remotely diagnose, monitor, configure and/or reprogram, as appropriate, the vehicles within a fleet. The tool would be used by subscribers to obtain real-time fleet characteristics, trend analysis and diagnostics, to perform manual, dynamic or rule based configuration, as well as allow fleet managers to provide real-time driver/fleet notification.

More specifically, the application service provider would provide a World Wide Web site where a fleet manager, using a computer and Web browser software, to remotely diagnose, monitor, configure, and/or reprogram the commercial vehicles for which they are responsible. Such fleet managers would include, for example, those responsible for overseeing a fleet of trucks for a commercial trucking or delivery company. Other users of the remote vehicle diagnostics, monitoring, configuring, and reprogramming tool would also include vehicle dealers, OEMs, and distributors who wish to obtain data concerning the performance of the vehicles within a fleet for "market intelligence" or "improved performance" purposes.

In an alternate embodiment, the remote vehicle diagnostics, monitoring, configuring and reprogramming tool of the present invention may be run, instead of on the global Internet, locally on proprietary equipment owned by the customers (i.e., the fleet managers, vehicle distributors, vehicle dealers and the like) as a stand alone software application. In yet another embodiment, users may access the remote vehicle diagnostics, monitoring, configuring and reprogramming tool of the present invention via direct dial-up lines rather than through the global Internet.

The remote vehicle diagnostics, monitoring, configuring, and reprogramming tool of the present invention would be utilized, as suggested above, by fleet manager users, for example, in order to facilitate vehicle parameter changes, track vehicle health, and/or receive indications of vehicle maintenance needs.

In an alternate embodiment, the remote vehicle diagnostics, monitoring, configuring and reprogramming tool of the present invention would be utilized by a vehicle component suppliers to re-calibrate any vehicle component, perform firmware downloads, perform component failure analysis, and determine wear characteristics.

In an alternate embodiment, the remote vehicle diagnostics, monitoring, configuring and reprogramming tool of the present invention would be utilized by vehicle manufacturers to analyze quality of components (and thus, suppliers) utilized in their manufacturing processes, and/or retrieve and manage warranty information.

In yet another embodiment, the remote vehicle diagnostics, monitoring, configuring and reprogramming tool of the present invention would be utilized by vehicle leasing companies to receive indications of vehicle maintenance needs, monitor vehicle use and abuse, and/or monitor lessee trip information.

In yet another alternate embodiment, the remote vehicle diagnostics, monitoring and reprogramming tool of the present invention would be utilized by vehicle dealers or vehicle repair facility personnel to perform proactive data

analysis, perform pre-arrival diagnostics, re-calibrate vehicle components, and/or perform firmware downloads.

The present invention is described in terms of the above examples. This is for convenience only and is not intended to limit the application of the present invention. In fact, after reading the following description, it will be apparent to one skilled in the relevant art(s) how to implement the following invention in alternative embodiments (e.g., to remotely manage different types and different aspects of vehicles--non-commercial or commercial, etc.).

The terms "user," "subscriber," "company," "business concern," and the plural form of these terms are used interchangeably throughout herein to refer to those who would access, use, and/or benefit from the remote vehicle diagnostics, monitoring and reprogramming tool of the present invention.

II.   System Architecture

Referring to FIG. 1, a block diagram illustrating the physical architecture of a total fleet logistics ("TFL") system 100, according to an embodiment of the present invention. FIG. 1 also shows network connectivity among the various components.

The TFL system 100 includes a plurality of users 102 (e.g., fleet managers, vehicle distributors, OEMs, vehicle dealers and the like) which would access to system 100 using a personal computer (PC) (e.g., an IBM™ or compatible PC workstation running the Microsoft® Windows 95/98™ or Windows NT™ operating system, Macintosh® computer running the Mac® OS operating system, or the like), running a commercially available Web browser. In alternative embodiments, users 102 may access TFL system 100 using any processing device including, but not limited to, a desktop computer, laptop, palmtop, workstation, set-top box, personal data assistant (PDA), and the like.

The users 102 would connect to the parts (i.e., infrastructure) of the TFL system 100 which are provided by the TFL application service provider (i.e., elements 106-124 of FIG. 1) via the global Internet 104. The connection to the

Internet 104, however, is through a firewall 106. The components of the TFL system 100 are divided into two regions--"inside" and "outside." The components in the "inside" region refer to those components that the TFL application service provider would have as part of their infrastructure in order to provide the tools and services contemplated by the present invention. As will be apparent to one skilled in the relevant art(s), all of components "inside" of the TFL system 100 are connected and communicate via a wide or local area network (WAN or LAN) running a secure communications protocol (e.g., secure sockets layer (SSL)). The firewall 106 serves as the connection and separation between the LAN, which includes the plurality of elements (e.g., elements 108-124) "inside" of the LAN, and the global Internet 104 "outside" of the LAN. Generally speaking, a firewall is a dedicated gateway machine (e.g., a SUN Ultra 10) with special security precaution software. It is typically used, for example, to service Internet 104 connections and dial-in lines, and protects the cluster of more loosely administered network elements hidden behind it from external invasion. Firewalls are well known in the relevant art(s) and firewall software is available from many vendors such as Check Point Software Technologies Corporation of Redwood City, CA.

TFL system 100 also includes two servers--an application server 108 and an onboard unit server ("OBU") 118.

The application server 108 is the "back-bone" (i.e., TFL processing) of the present invention. It provides the "front-end" for the TFL system 100. That is, application server 108 includes a Web service 110 which is a typical Web server process running at a Web site which sends out Web pages in response to Hypertext Transfer Protocol (HTTP) requests from remote browsers (i.e., subscribers 102 of the TFL application service provider). More specifically, a Web server 112 provides graphical user interface (GUI) "front-end" screens to users 102 of the TFL system 100 in the form of Web pages. These Web pages, when sent to the subscriber's PC (or the like), would result in GUI screens being displayed. In an embodiment of the present invention, the server 112 would be implemented using a Netscape Enterprise or compatible Web server, an Apache

web server or the like. Connected to the server 112 is an application server 114 which facilitates the data and commands between a repository database 116 and the Web pages on Web server 112. In an embodiment of the present invention, the server 114 would be an Oracle application server.

Also included in the application server 108 is a TFL repository database 116. Database 116, in an embodiment of the present invention, is a Sun E250 machine running the Oracle 8i RDBMS (relational database management server) software. The database 116 is the central store for all information within the TFL system 100 and also stores Web page executable code (e.g., PL/SQL and HTML).

The OBU server 118 is responsible, generally, for routing data between the smart device onboard units 130 within each vehicle (explained in detail below) and the application server 108. The OBU server 118 includes three software modules, implemented in a high level programming language such as the C++ programming language--a dispatcher 120, a communications service 122, and a conversion service 124. The dispatcher 120 is a software module resident on the OBU server 118 and is responsible for serving as an intermediary to route messages between the remaining two components of the OBU server 118 (i.e., the communications service 122 and the conversion service 124).

The communications service 122 is a module that contains software code logic that is responsible for handling in-bound and out-bound vehicle data and commands. As will be described in more detail below, the communications service 122 is configured for the specific means of mobile communications employed within TFL system 100 (e.g., satellite or terrestrial wireless).

The conversion service 124 is a module that contains software code logic that is responsible for converting raw vehicle data (i.e., telemetry) into human-readable format, and vice-versa. In an embodiment of the present invention, the conversion service 124 module includes a relational database implemented in Microsoft® Access or the like which stores telemetry data definitions for a plurality of vehicle makes, models, and associated components. Such definitions would include vehicle component masks, bit length, and data stream order definitions for various vehicle (and component) manufacturers in order to

perform the binary (raw) data conversion into human-readable form, and vice-versa.

TFL system 100 also includes an administrative workstation 134. This workstation can be used by personnel of the TFL application service provider to upload, update, and maintain subscriber information (e.g., logins, passwords, etc.) and fleet-related data for each of the users 102 that subscribe to the TFL system 100. The administrative workstation 134 may also be used to monitor and log statistics related to the application server 108 and system 100 in general. Also, the administrative workstation 134 may be used "off-line" by subscribers 102 of the TFL system 100 in order to enter configuration data for supported controllers 132, etc. within their fleet(s). This data is eventually stored in TFL repository database 116.

TFL system 100 also includes a plurality of vehicles 128 (i.e., the "fleet" being remotely diagnosed, monitored and/or reprogrammed). (FIG. 1 shows only one vehicle 128 for ease of explanation herein.) Within each vehicle is a smart device onboard unit 130, explained in more detail below. In an embodiment of the present invention, the onboard units 130 have access to a plurality of controllers or discrete measurement points 132 (shown as controllers 132a-n in FIG. 1) found within the vehicle 128 (e.g., brake, engine, transmission, and various other vehicle electrical component controllers). Such access is though the vehicle data bus (not shown) of each of the vehicles 128. Further, the onboard units 130 include transceivers that communicate with a communications service provider 126. Like the communications service module 122, the onboard units 130 are configured for the specific means of wireless mobile communications employed within TFL system 100 (e.g., satellite or terrestrial wireless).

More detailed descriptions of the TFL system 100 components, as well their functionality, are provided below.

III. On Board Units

Referring to FIG. 2A, a block diagram of the physical architecture of the onboard unit 130, in a preferred embodiment of the present invention, is shown. The onboard unit 130 handles communications between the vehicle controllers 132 and the remainder of the TFL system 100.

In a preferred embodiment of the present invention, the onboard unit 130 is a small (e.g., 5" x 6" x 2") computer board which contains a 32-bit RISC architecture central processing unit (CPU) 202 such as the Intel® Strong ARM 32-bit chip, a 4 megabyte (MB) random access memory (RAM) 204, a 4MB flash memory 206, a power supply 208, and a compact flash interface memory 210.

Further, onboard unit 130 also includes a user interface channel ports 212 and a vehicle interface channel ports 214. In an embodiment of the present invention, the user interface channel ports 212 contain interface modules for several wire and wireless mobile communications standard devices such as universal serial bus (USB), standard parallel ports, standard serial ports, satellite communications, code division multiple access (CDMA), time division multiple access (TDMA), the Bluetooth® wireless standard chip, intellect data bus (IDB), and the like. This would allow the TFL application service provider to utilize several of the available providers 126 to communicate with vehicles 128 in their subscriber's fleets.

In an embodiment of the present invention, the vehicle interface channel ports 214 contain interface modules for several standard automotive application program interfaces (API's). Such API's include *Serial Data Communications Between Microcomputer Systems in Heavy-Duty Vehicle Applications*, Document No. J1708, Society of Automotive Engineers (SAE) of Warrendale, PA (October 1993); *Joint SAE/TMC Electronic Data Interchange Between Microcomputer Systems in Heavy-Duty Vehicle Applications*, Document No. J1587, SAE (July 1998); and *Recommended Practice for Truck and Bus Control and Communications Network*, Document No. J1939, SAE (April 2000); all of which are incorporated herein by reference in their entirety. Other such API's include

SAE's onboard diagnostic system (OBD) II standard and several vehicle manufacturer specific/proprietary interfaces and discrete measurement point interfaces.

Referring to **FIG. 2B**, a block diagram of the software architecture of the onboard unit 130, in a preferred embodiment of the present invention, is shown. Onboard unit 130 contains three main software modules, implemented in a high level programming language such as the C++ programming language, and executing on the CPU 202. These modules include a command server module 210, a plurality of application specific modules 220 (shown as application specific modules 220a-n), and a data parser/requester module 230.

The command server module 210 contains software code logic that is responsible for handling the receiving and transmitting of the communications from the provider 126 and relays such data to either the data parser/requester module 230 or to one of the application specific modules 220, as applicable.

The application specific modules 220 (one for each manufacturer specific controller 132 within the vehicle) each contain software code logic that is responsible for handling interfacing between the command server module 210 to the vehicle data bus 240 (via data parser/requestor module 230) for application specific (i.e., manufacturer specific) parameter readings, alerts, configuration or reprogramming data (as explained in detail below).

The data parser/requester module 230 contains software code logic that is also responsible for handling direct interfacing between the command server module 210 to the vehicle data bus 240 for non-application specific (i.e., "generic" SAE J1708 or SAE1939 discrete measurement points) parameter readings, alerts, configuration or reprogramming data (as explained in detail below).

In an embodiment of the present invention, the onboard unit 130 is designed to be compliant with the SAE's *Joint SAE/TMC Recommended Environmental Practices for Electronic Equipment Design (Heavy-Duty Trucks)*, Document No. J1455 (August 1994) standard, which is incorporated herein by reference in its entirety, because it will be a component included (or installed)

within vehicles 132. That is, the onboard unit 130 is physically mounted on the vehicle 128, electrically coupled to the vehicle data bus 240 via the wiring harness of the vehicle 128, and packaged in a manner that resists environmental seepage of dirt and moisture, as well as withstands operational vibration. Further,

5   the onboard unit 130 must be built to withstand, in a preferred embodiment, industrial temperature ranges of -40 to 85 degrees centigrade.

In an alternate embodiment of the present invention, the onboard unit 130 would include a global positioning (GPS) receiver component, which would allow the TFL system 100 to provide location-based logistical management

10   features to users 102.

More details of the onboard unit 130 architecture and functionality are provided below in connection with the description of the TFL system 100 operation.

IV.   Detailed Example of System Operation

15   Referring to FIG. 3, a flow chart of a sample control flow 300, according to an embodiment of the present invention, is shown. More specifically, control flow 300 depicts a fleet manager user 102 reprogramming a fleet vehicle parameter with reference to the elements of TFL system 100 described above with reference to FIG. 1. (Also see FIG. 6 described below.) Control flow 300

20   begins at step 302, with control passing immediately to step 304.

In step 304, the user 102 enters their password in order to login into the TFL system 100. Such login would be provided by a Web page sent out over the Internet 104 (and accessed by user 102 using a PC or the like) by Web service 110. Subscriber information would be kept by the TFL application service

25   provider in the TFL repository database 116.

After the user is logged in, in step 306, the user then enters their vehicle list selection. The vehicle choices (i.e., entire fleet(s), division(s) of vehicles within a fleet, or specific individual vehicles) available for selection are stored for each subscriber in the TFL repository database 116. Once presented with a GUI

of available vehicles, in step 308, the user 102 would then enter the parameter(s) (e.g., max cruise speed) they would like to reprogram on the specific vehicle(s) selected in step 306. In step 310, the user 102 would enter the new setting(s) (e.g., 55 MPH) for the selected parameter(s).

In step 312, the application server 108 receives the settings and translates the reprogramming request into a list of commands--one command for each vehicle--and forwards these commands to the dispatcher module 120 located on the onboard unit (OBU) server 118. In step 314, the dispatcher 120 forwards each command to the conversion service 124. In step 316, the conversion service 124 translates the user entered setting(s) (e.g., "55 MPH") to a binary format understandable to the onboard unit 130 such that it can process the command according to the requirements of the targeted vehicle controller 132. This translation is facilitated by the relational database (as described above) located within the conversion service 124. Once translated, the command (now in binary) is sent back to the dispatcher 120.

In step 318, the conversion service 124 forwards the command to the communications service 122. In step 320, the communications service 122 further encodes and compresses the command (for efficiency of transmission), and routes the command, (passing the firewall 106 and) via the Internet 104, to the communications provider 126. In step 322, the communications provider 126 forwards the command to the onboard unit 130 on the vehicle 128.

As mentioned above, step 322 may be accomplished, depending on the embodiment of the present invention (i.e., according to the provider 126 selected by or available to the TFL application service provider), via any wire or wireless mobile communications standard such as USB, parallel ports, serial ports, satellite communications, CDMA, TDMA, the Bluetooth® wireless standard, IDB, and the like.

In an embodiment of the present invention, more than one communication service provider 126 (and thus more than one means of mobile communications) would be utilized by the TFL application service provider in order to maximize the number of different vehicles 128 belonging to different subscribers 102 that

may be diagnosed, monitored and/or reprogrammed by the TFL system 100. Consequently, the OBU server 118 would contain multiple communications service 122 modules, each configured for specific communication service provider 126.

In step 324, the command is received by the command server module 210 executing on the CPU 202 of the onboard unit 130. In step 326, the command is forwarded to the vehicle data bus 240 by the data parser requester module 230 executing on the CPU 202 of the onboard unit 130. The command thus finally reaches the appropriate controller 132 within the vehicle 128. Control flow 300 then ends as indicated by step 328.

As will be apparent to one skilled in the relevant art(s) after reading the above, an acknowledgment of the reprogramming command from the vehicle 128 to the user 102 would flow in the reverse direction from control flow 300. Further, the acknowledgment would be stored in database 116 for the user 102 to (later) retrieve.

It should be understood that control flow 300, which highlights the reprogramming functionality of TFL system 100, is presented for example purposes only. The software architecture of the present invention is sufficiently flexible and configurable such that users 102 may navigate through the system 100 in ways other than that shown in **FIG. 3**.

V.   Graphical User Interface

As mentioned above, the application server 108 will provide a GUI for users 102 (e.g., fleet managers, vehicle distributors, OEMs, vehicle dealers and the like) to enter inputs and receive the outputs as described, for example, in control flow 300. In an embodiment of the present invention, the GUI screens of the present invention may be classified into three categories: alerts (e.g., threshold alerts, tamper warnings, etc.), parameter readings, and reprogramming. **FIGS. 4-6**, presented below, show examples GUI screens reflecting these three categories

Insert A1

respectively. They also highlight the functionality and features of TFL system 100 in general.

Referring to **FIG. 4A**, a "set alert" GUI screen 410 with representative data, according to an embodiment of the present invention, is shown. Screen 400 includes a column 402 labeled "Vehicle Unit ID" which indicates the vehicles within a fleet the user 102 has previously selected to receive alerts for. Screen 400 includes a column 404 labeled "Description" which indicates the type of vehicle 128 corresponding the Vehicle Unit ID in column 402. Screen 400 also includes a column 406 labeled "T. Codes" which is a check box the user 102 can select to indicate that they wish to track alert codes for all available parameters within a specific vehicle 128. Lastly, screen 400 includes a column 408 labeled "Tamper" which is a check box the user 102 can select to indicate whether they wish to track whether any parameter within a specific vehicle 128 has been physically tampered with.

Referring to **FIG. 4B**, a "view alert" GUI screen 410 with representative data, according to an embodiment of the present invention, is shown. Screen 410 includes a column 412 labeled "Reading Date/Time" which indicates the actual date and time a particular alert was generated for a particular vehicle specified in a column 414 labeled "Vehicle ID." In a column 416, the parameter name (e.g., vehicle speed limit) for which the alert was generated is displayed. Screen 410 also includes a column 418 labeled "Alert Value," where a description of the alter is displayed.

Referring to **FIG. 5A**, a "select parameter" GUI screen 500, according to an embodiment of the present invention, is shown. Screen 500 includes four categories 502a-d of parameters a user 102 may select. Within each category 502, there are specific vehicle parameters 504a-d that the user 102 may choose from. Selected parameters 504 or categories of parameters 502 will result in the TFL system 100 system obtaining these parameter readings from each of the vehicles 128 that the user 102 has previously selected.

Referring to **FIG. 5B**, a "select parameter transactions" GUI screen 510 with representative data, according to an embodiment of the present invention,

is shown. Screen 510 includes a column 512 labeled "Transaction Description." This column indicates the names of the different transactions created by one or more users 102 which manage the same fleet of vehicles. In an embodiment of the present invention, a "transaction" is a section of different parameter categories 502 and/or specific vehicle parameters 504 selected by a user 102 using screen 500 and saved in the TFL system 100 using a "transaction" name shown in column 512 of screen 510. A column 513 indicates the ID (i.e., login name) of the particular user 102 which created the transaction. A column 514 indicates the date that the user 102 created the transaction. A column 516 labeled "Param Profile Requested" indicates the category 502 of parameters that the user 102 selected in GUI screen 500 for the corresponding transaction. A column 518 allows the user 102 to select the transactions they would like to view for the specific vehicles 128 previously selected.

Referring to **FIG. 5C**, a "view parameter results" GUI screen 520, according to an embodiment of the present invention, is shown. Screen 520 includes a column 522 labeled "Vehicle Unit ID" which indicates the vehicles within a fleet the user 102 has previously selected to receive parameter readings from. Screen 520 also includes several parameter reading columns 524 which indicate the parameter values read from the selected vehicles 128 and correspond to the transaction selected by a user 102 using the select buttons in column 518 on screen 510.

Referring to **FIG. 6A**, an "enter parameter values for reprogramming" GUI screen 600, according to an embodiment of the present invention, is shown. Screen 600 includes a column 602 labeled "Vehicle Unit ID" which indicates the vehicles within a fleet the user 102 has previously selected to reprogram. (See control flow 300 described above with reference to **FIG. 3**.) Screen 600 includes a column 604 labeled "Description" which indicates the type of vehicle 128 corresponding the Vehicle Unit ID in column 602. Screen 600 also includes a column 606 labeled "Current Setting" which indicates the current value of the previously selected parameter that user 102 desires to reprogram (i.e., change). Lastly, screen 600 includes a column 608 labeled "New Setting" which is an

input box where the user can enter a new value for the previously selected vehicle 128 parameter.

Referring to **FIG. 6B**, a "view reprogramming results " GUI screen 610, according to an embodiment of the present invention, is shown. Screen 610 includes a column 612 labeled "Vehicle" which indicates the vehicles 132 within a fleet the user 102 has previously selected to reprogram. A column 614 indicates the name of the previously selected vehicle parameter for which status information is now being viewed by user 102. A column 616 indicates the date and time that the user 102 submitted the reprogramming request using screen 600. A column 618 labeled "Current" indicates the present value (at last reading and presently stored in repository 116) for the corresponding vehicle parameter shown in column 614. A column 620 labeled "Requested" indicates the new reprogrammed value requested by user 102 using column 608 of screen 600. Screen 610 also includes a column 622 labeled "Status" which indicates the current status (as read from the vehicle 128) of the reprogramming command sent by the TFL system 100.

It should be understood that the screens shown in this section (i.e., **FIGS. 4-6**), which highlights the functionality of TFL system 100, are presented for example purposes only. The software architecture (and thus, GUI screens) of the present invention is sufficiently flexible and configurable such that users 102 may navigate through the system 100 in ways other than those shown in **FIGS. 4-6**. Further, the information described therein can be presented to the user 102 in ways other than shown in **FIGS. 4-6**.

In an embodiment of the present invention, reprogramming commands to be sent to specific vehicles 128 and parameter readings to be read from specific vehicles 128 can be scheduled by the TFL system 100. That is, the user 102 may specify, for example, pre-defined time periods that parameter readings should be taken for specific vehicles within a fleet. Such pre-defined time periods can be hourly, daily, x times per day, weekly, y times per week, monthly, etc.

VI.   *Example Implementations*

The present invention (i.e., TFL system 100, onboard unit 130, control flow 300, and/or any part(s) thereof) may be implemented using hardware, software or a combination thereof and may be implemented in one or more computer systems or other processing systems. In fact, in one embodiment, the invention is directed toward one or more computer systems capable of carrying out the functionality described herein. An example of a computer system 700 is shown in FIG. 7. The computer system 700 includes one or more processors, such as processor 704. The processor 704 is connected to a communication infrastructure 706 (e.g., a communications bus, cross-over bar, or network). Various software embodiments are described in terms of this exemplary computer system. After reading this description, it will become apparent to a person skilled in the relevant art(s) how to implement the invention using other computer systems and/or computer architectures.

Computer system 700 can include a display interface 705 that forwards graphics, text, and other data from the communication infrastructure 702 (or from a frame buffer not shown) for display on the display unit 730.

Computer system 700 also includes a main memory 708, preferably random access memory (RAM), and may also include a secondary memory 710. The secondary memory 710 may include, for example, a hard disk drive 712 and/or a removable storage drive 714, representing a floppy disk drive, a magnetic tape drive, an optical disk drive, etc. The removable storage drive 714 reads from and/or writes to a removable storage unit 718 in a well known manner. Removable storage unit 718, represents a floppy disk, magnetic tape, optical disk, etc. which is read by and written to by removable storage drive 714. As will be appreciated, the removable storage unit 718 includes a computer usable storage medium having stored therein computer software and/or data.

In alternative embodiments, secondary memory 710 may include other similar means for allowing computer programs or other instructions to be loaded into computer system 700. Such means may include, for example, a removable

storage unit 722 and an interface 720. Examples of such may include a program cartridge and cartridge interface (such as that found in video game devices), a removable memory chip (such as an EPROM, or PROM) and associated socket, and other removable storage units 722 and interfaces 720 which allow software

5   and data to be transferred from the removable storage unit 722 to computer system 700.

Computer system 700 may also include a communications interface 724. Communications interface 724 allows software and data to be transferred between computer system 700 and external devices. Examples of communications

10   interface 724 may include a modem, a network interface (such as an Ethernet card), a communications port, a PCMCIA slot and card, etc. Software and data transferred via communications interface 724 are in the form of signals 728 which may be electronic, electromagnetic, optical or other signals capable of being received by communications interface 724. These signals 728 are provided to

15   communications interface 724 via a communications path (i.e., channel) 726. This channel 726 carries signals 728 and may be implemented using wire or cable, fiber optics, a phone line, a cellular phone link, an RF link and other communications channels.

In this document, the terms "computer program medium" and "computer usable medium" are used to generally refer to media such as removable storage

20   drive 714, a hard disk installed in hard disk drive 712, and signals 728. These computer program products are means for providing software to computer system 700. The invention is directed to such computer program products.

Computer programs (also called computer control logic) are stored in

25   main memory 708 and/or secondary memory 710. Computer programs may also be received via communications interface 724. Such computer programs, when executed, enable the computer system 700 to perform the features of the present invention as discussed herein. In particular, the computer programs, when executed, enable the processor 704 to perform the features of the present

30   invention. Accordingly, such computer programs represent controllers of the computer system 700.

In an embodiment where the invention is implemented using software, the software may be stored in a computer program product and loaded into computer system 700 using removable storage drive 714, hard drive 712 or communications interface 724. The control logic (software), when executed by the processor 704, causes the processor 704 to perform the functions of the invention as described herein.

In another embodiment, the invention is implemented primarily in hardware using, for example, hardware components such as application specific integrated circuits (ASICs). Implementation of the hardware state machine so as to perform the functions described herein will be apparent to persons skilled in the relevant art(s).

In yet another embodiment, the invention is implemented using a combination of both hardware and software.

VII.   Conclusion

While various embodiments of the present invention have been described above, it should be understood that they have been presented by way of example, and not limitation. It will be apparent to persons skilled in the relevant art(s) that various changes in form and detail can be made therein without departing from the spirit and scope of the invention. Thus the present invention should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

60/354,673

NEXIQ™

TECHNOLOGIES

xVDS White Paper
Revision #   1
Revision Date   8/30/01

White Paper
xVDS
Written by: Kevin A. Williams

Table of Contents

Introduction

As vehicle and vehicle component manufacturers continue the trend toward the increased usage of electronic controls, the need for more and varied vehicle-interactive software applications becomes apparent.

The development of vehicle-interactive applications, however, is not an easy task. A design and development team can easily be consumed by the considerations of the vehicle interactions themselves instead of focusing on the target application.

xVDS addresses these issues by providing an efficient, portable, reliable and extensible infrastructure, leaving the system developer free to address application-level issues.

Vehicle-Interactive applications

When considering a vehicle-interactive application (either diagnostic or telematic) the same basic rules apply: You must establish communications with the vehicle controller; construct appropriate messages to send to the controller; and process the messages the controller sends to you. (Figure 1).

Vehicle   Engine   Application
                           Controller
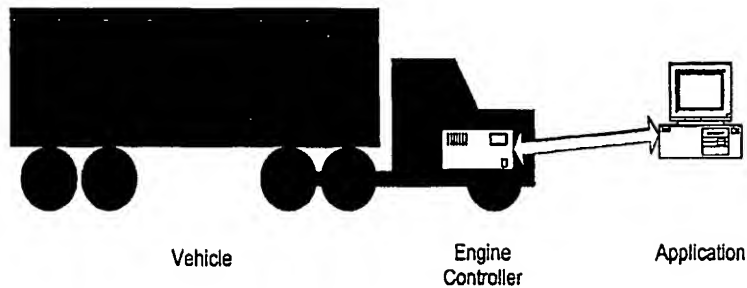
Figure 1. Simple Vehicle-Interactive System

Systems are typically written around a specific vehicle communications protocol (i.e. 1708, 1939, etc) design/development commences from there. Only as the system design evolves does it become apparent that vehicle-interactive applications are not as simple as first thought (Figure 2).
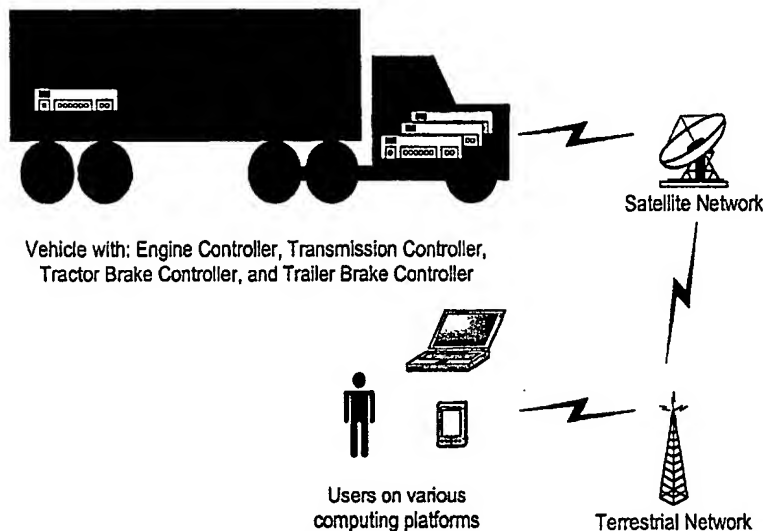

Figure 2. Increased Vehicle-Interactive Complexity

The development of a vehicle-interactive system is often complicated by a variety of factors:

- It may be necessary to support more than one type of platform, such as embedded computers, Windows CE devices, and PalmOS devices.

- A single vehicle protocol may not be sufficient to communicate with every controller installed on the vehicle.

- The application itself may need to be distributed between the vehicle, various servers, and the final display platform.

xVDS addresses these issues.

xVDS

xVDS provides an standard infrastructure for creating vehicle-interactive applications, without locking the system into a single protocol, platform, or communication system.

The framework is data-driven. Much of the functionality required to support vehicle-interactive applications is implemented by the framework acting upon the vehicle database – relieving the application developer of writing this code.

The framework is cross-platform, providing the same services on differing platforms. The framework is also easily ported to new platforms, as an OS Thin Layer abstracts the operating system and hardware dependencies.

The modular approach of the framework allows full customization. Functionality components can be replaced with new algorithms or removed entirely from the system based on the desired behavior.

Architecture

xVDS is designed to support multiple vehicle controllers communicating over multiple protocols on a variety of platforms connected to a variety of user interfaces.
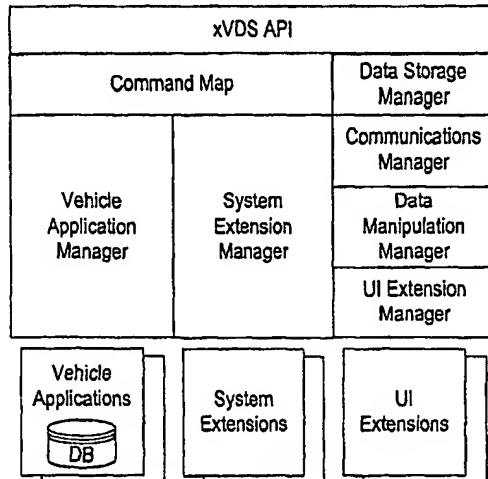
xVDS API		
Command Map		Data Storage Manager
Vehicle Application Manager	System Extension Manager	Communications Manager
		Data Manipulation Manager
		UI Extension Manager
Vehicle Applications DB	System Extensions	UI Extensions

Figure 3. xVDS Architecture

xVDS includes the following components (Figure 3):

xVDS API

The xVDS API provides a standard interface for Vehicle Applications and System Extensions to make use of the framework.

Command Map

The Command Map allows System Extensions and Vehicle Applications to register functions and make them available to all other components of the system.

System Extension Manager

The System Extension Manager manages the loading, initialization, shut-down, unloading and reporting of System Extensions.

Vehicle Application Manager

The Vehicle Application Manager manages the loading, initialization, shut-down, unloading and reporting of Vehicle Applications.

Communications Manager

The Communications Manager provides a protocol independent means of communicating with the vehicle controller(s) involved in the application.

Data Storage Manager

The Data Storage Manager is responsible for maintaining current and historical values for data collected from vehicle controllers.

Data Manipulation Manager

The Data Manipulation Manager is responsible for calculating real-world values and units from the raw data collected from vehicle controllers.

System Extensions

System Extensions provide a standard method for adding functionality to the framework without modifying the system itself. Using system extensions keeps the system both scalable and distributable.

Vehicle Applications

Vehicle Applications contain the actual application functionality. Business logic is incorporated into both a vehicle database and application code, which work together within the framework to form a complete application.

User Interface Extension Manager

The User Interface Extension Manager manages the loading, initialization, shut-down, unloading and reporting of User Interface Extensions.

User Interface Extensions

User interface extensions allow the framework to take input from and display results on a variety of platforms.

Features

xVDS supports the following features:

Common API

The xVDS API abstracts the differences between vehicle controllers, protocols, and transport layers.

Data Driven Operation

Data Driven Operation allows the application developer to concentrate design and implementation efforts on the business requirements of the application instead of spending coding time on vehicle-interaction.

Extensability

System extensions allow functionality to be added at any time, without modifying (or revalidating) the base system.

Portability

xVDS is designed to be easily ported between platforms. An OS Thin Layer isolates the framework from OS differences and allows for ease of porting.

The framework is designed to be deployed as DLLs or Shared Libraries on platforms that support these, and as statically linked libraries in other cases.

Scalability

System extensions may be added or removed based upon the amount and type of processing required on the target platform. For example – A system that records vehicle information for later review doesn't need to carry the weight of the full implementation; however, such a system could be scaled to process and respond to certain conditions by adding the appropriate system extensions and vehicle application module(s).

Distributed processing capability

As required by the application, functionality may be distributed across multiple systems. Components of xVDS may reside on the vehicle, at the user interface level, or anywhere in between.

Summary

The creation of vehicle-interactive applications can appear simple, but quickly become complex as the following issues come into play:

- multiple controllers
- multiple protocols
- multiple platforms
- distributed processing.

xVDS provides an extensible, portable, reliable and efficient infrastructure for creating vehicle-interactive systems. Use of the framework can reduce time to market and product development cost by freeing the development team to concentrate on the application or service being provided rather than struggle with the complexities of vehicle interaction.

Change Log

Revision	Sections Changed	Change Description	Date
1	All sections	Initial creation.	8/30/01

(Current revision 1; last revision date: 8/30/01 Please do not disturb this line of text; it supports automatic version numbering.)

This document is archived in VSS under: $/NEXIQ/xVDS/Docs.

60/351,165

DSI Wireless Communication Framework White Paper

Version Table

Name	Date	Version	Comments
Andrew Smith	1/8/2001	1.0	Initial Version
Greg Dils	1/12/2001	1.1	Added features
Greg Dils	1/23/2001	1.2	More comments
Greg Dils	2/2/2001	1.3	Changed title

This document was created with Microsoft Word 2000.

This document is archived in SourceSafe in *$/WCF/Docs.*

Table of Contents

Table of Figures

1 Introduction

The growing availability of wireless communication infrastructures and low cost mobile computing platforms is enabling a great variety of new enterprise and service applications. These applications are driven by the business opportunities of providing new services, such as in-vehicle messaging, and reducing the cost of existing services while improving service to the customer, by optimizing the application of resources based on real-time access to data.

The development of communication-based applications, however, is not an easy task. A design and development team can easily be consumed by the considerations of wireless communications instead of focusing on the target application.

The DSI Wireless Communication Framework addresses these issues by providing an efficient, portable, reliable and extensible infrastructure, leaving the system developer free to address application-level issues.

2 The Trouble with Mobile Communications

When considering an enterprise application or business service offering that relies on wireless communication, the initial system concept seems straightforward enough... you have mobile users that communicate with the enterprise application via a wireless communication service (Figure 1).
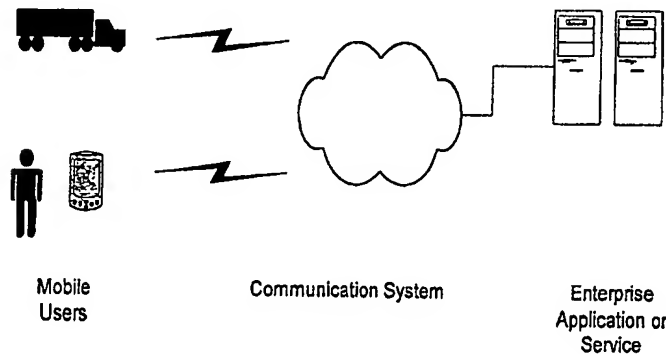

Mobile Users

Communication System

Enterprise Application or Service

Figure 1. Simple Wireless System

Typically a communication provider is selected, and design and development commence. Only as the system design evolves does it become apparent that wireless communications are not as simple as first thought (Figure 2).
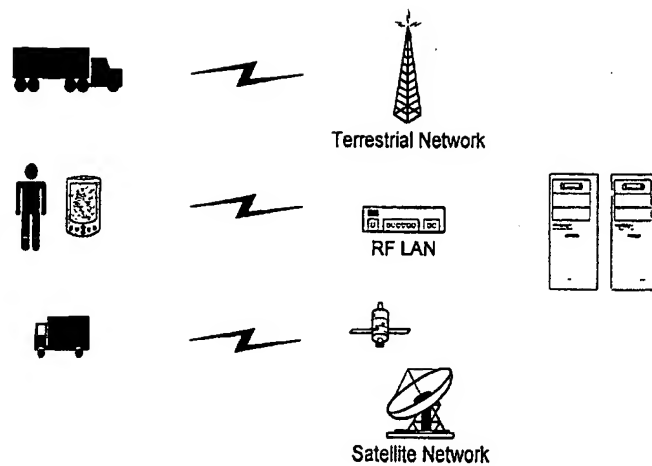
Figure 2. Increased Wireless Complexity

The development of a communication-based system is often complicated by a variety of factors:

- It may be necessary to support more than one type of mobile platform, such as embedded computers, Windows CE devices, and PalmOS devices.

- A single wireless service or technology may not provide the messaging cost structure or geographical coverage to support all the desired system users. Multiple services might be used to provide for dynamic balancing between messaging cost and message timeliness.

- Different wireless services and technologies have different APIs or may require custom interfaces for the target environments.

- It can be difficult and time consuming to create the infrastructure and protocol that provides the desired messaging services to the application.

- Messaging capabilities differ across services and technologies.

The DSI Wireless Communication Framework addresses these issues.

3 The Wireless Communication Framework

The DSI Wireless Communication Framework provides an infrastructure for creating communication-based applications, without locking the system into a single communication provider or target platform.

The framework is cross-platform, providing the same messaging services on differing mobile platforms. The framework is also easily ported to new platforms, as an OS Thin Layer abstracts the operating system and hardware dependencies.

The framework provides a standard messaging API regardless of platform and wireless service. It allows the use of multiple transports (communication services) and includes optional capability for least-cost routing and multi-packet messaging.

The modular approach of the framework allows full customization. Functionality components can be replaced with new algorithms or removed entirely from the system based on the desired behavior.

3.1 Architecture

The Wireless Communication Framework is designed to support multiple mobile devices communicating with a host system consisting of one or more communication and application servers (Figure 3). Multiple applications can share the Wireless Communication Framework, on both the mobile and server ends of the link. The actual communication services are implemented using Transport Modules, which can be added or removed to enable specific communication providers.
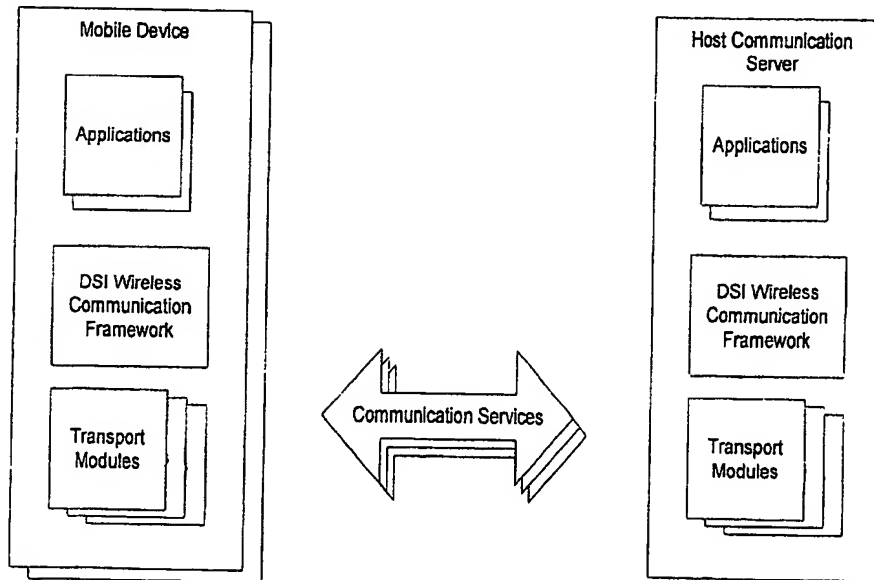

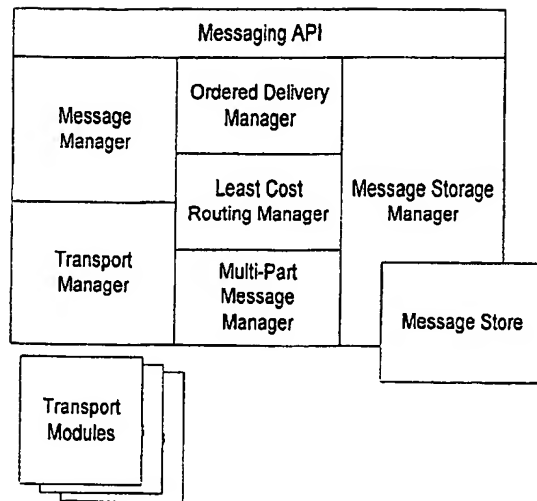Figure 3. Wireless Communication Framework System Architecture

Figure 4. Inside the Wireless Communication Framework

The Wireless Communication Framework includes the following components (Figure 4):

Messaging API

The Messaging API is the set of functions and data structures exposed to the application to allow it to perform messaging.

Message Manager

The Message Manager manages the acceptance of messages from the application and the delivery of messages to the application. It also manages the reliable delivery of messages.

Transport Modules

Transport modules are the individual modules that implement an interface with specific communication providers and technologies. These abstract the differences between communication services.

Transport Manager

The transport manager is responsible for identifying the available transport modules and (in conjunction with the Least Cost Routing Manager) selecting a transport over which each message will be transferred.

Message Storage Manager

The Message Storage Manager is responsible for keeping the queue(s) of messages that are pending for send or receive.

Message Store

The Message Store is the implementation of message storage. It may be modified according to the storage features available on the platform and the desired reliability of the end system.

Ordered Delivery Manager

The Ordered Delivery Manager is an optional component that can provide ordered delivery service for those applications that require it.

useful when the application sends information that depends on the state of information sent earlier; such as sending database edit deltas.

Multi-Part Messages

A Wireless Communication Framework installation can include optional support for multi-part messages. This support allows messages larger than transport-native messages to be broken down for transmission and reassembled at the receiving end.

Message Priority

A message priority can be specified for each message. Message priority is used as the primary factor in determining transport selection in least cost routing. Message priority can also be used to determine which messages should be sent first if multiple messages are ready to be sent at the same time. On the receiving side, priority will be used to determine the order in which messages are sent to the destination application.

Broadcast

Messages sent from the server can be sent to multiple mobile devices (broadcast). If the selected transport(s) support broadcast messaging, the transport-specific broadcast can be used if this reduces bandwidth or cost.

Destination Application

For systems where more than one application is making use of the Wireless Communication Framework, a destination application can be specified that allows routing from the messaging system to the correct target application.

Delivery Confirmation

For messages sent with reliable delivery enabled, an optional delivery confirmation notification can be established. This can be used in lieu of application response messaging in those cases where it is sufficient to take action when delivery to the recipient's messaging system has been confirmed.

Encryption

Messages can optionally be encrypted over the entire communication route. The Wireless Communication Framework will support any current or future method of encryption.

Compression

To limit the bandwidth required for transmission, a message can be compressed. This optional functionality will allow any common compression algorithm to be used. As custom or improved algorithms are created, the Wireless Communication Framework can incorporate them into a system.

Packaging

The transmission of small messages over transport mechanisms that support larger message sizes may result in a higher message cost due to the overhead involved in acknowledgements. The Wireless Communication Framework supports the packaging of multiple small messages into a single larger message. This allows the acknowledgement cost to occur just once for several application messages.

3.3 Extensibility

The Wireless Communication Framework is extensible in several ways. The extension interfaces allow the system/application designer to customize and extend the framework as needed for the target system capabilities and behavior. Extension capabilities include the following:

Message Store

The Message Store provides the storage necessary for messaging functions, including persistence (e.g. retaining message information over a reset), reliable delivery, ordered delivery and multi-part messaging. The message store can be customized to according to platform capabilities and system requirements.

Transport Modules

Transport modules provide the interface to specific messaging services and providers. New transport modules can be written to support additional services and providers.

Least Cost Routing

The least cost routing manager can be customized to provide sophisticated routing and escalation logic as needed for the target system.

Compression and Encryption

As new standards and methods are developed for message compression and encryption, the Wireless Communication Framework can be extended to incorporate these changes. New compression and encryption modules could also be written to support custom or proprietary methods.

3.4 Portability

The mobile framework is designed to be easily ported between platforms. An OS Thin Layer isolates the framework from OS differences and allows for ease of porting. The Message Store abstracts the file system, memory structures, or database in which message information is stored.

The framework is designed to be deployed as DLLs or Shared Libraries on platforms that support these, and as statically linked libraries in other cases.

The server side framework makes use of COM and can be integrated with COM+ on Windows 2000 if the transactional and security features of COM+ are desired. This provides a robust integration with Enterprise Systems requiring the ability to send and receive messages within the context of database transactions.

4 Using the Framework

A typical project using the framework would take the following steps:

Mobile Development

1. Select the mobile features and transports based on the system requirements and design.

2. Port the mobile framework to the target platform. Application development can begin once the messaging capabilities have been specified and the API validated for the target.

3. Port or write the Message Store for the target platform.

4. Port or write the Transport Modules for the target platform.

5. Customize least cost routing if necessary.

6. Integrate with the application.

Server Development

1. Select features and transports based on the system requirements and design. Application development can begin once the messaging capabilities have been identified.

2. Adapt the Message Store to the server capabilities, if necessary.

3. Write any new Transport Modules.

4. Customize least cost routing if necessary.

5. Integrate with the application.

5 Summary

The creation of a wireless-enabled application or service can appear simple but quickly become complex as the issues come into play of message services, geographic diversity, message costs, and multiple service providers.

The DSI Wireless Communication Framework provides an extensible, portable, reliable and efficient infrastructure for creating wireless-enabled systems. Use of the framework can reduce time to market and product development cost by freeing the development team to concentrate on the application or service being provided rather than struggle with the complexities of wireless messaging.

Least Cost Routing Manager

The Least Cost Routing Manager is responsible for selecting a transport for each message based on cost and timeliness. This module can be replaced by a custom version to provide additional routing behavior. It can also be omitted from the system if no such functionality is desired.

Multi-Part Message Manager

The Multi-Part Message Manager is an optional component that provides the capability to break up and reassemble large messages when the maximum message size for a system exceeds the size limit of the transports in use. This module could be removed from the system in cases where this functionality is provided by the underlying transport, or where the messages sent will never exceed the transport message size limit.

3.2 Features

The Wireless Communication Framework supports the following features:

Common Messaging API

The Messaging API abstracts the differences between transport providers and technologies and allows applications to be written independently of the services or providers selected.

Multiple Transports

Multiple communication services and providers are supported. This allows the use of differing services and providers to allow greater geographic coverage and to balance messaging cost and timeliness. Transports can include those with no per-message cost such as direct connections and wireless LANs. This can be used to support development and testing, as well as to provide low-cost messaging at specially equipped sites.

Least-Cost Routing

When multiple transports are installed simultaneously on a mobile device, least cost routing can be used to determine the transport(s) over which messages should be sent based on timeliness and cost drivers. Least cost routing can be customized to provide additional capabilities.

Reliable Transport Service

Messages can be sent with the reliable transport option selected. The Wireless Communication Framework then uses internal acknowledgements to guarantee message delivery. This option can be disabled when the application is using an end-to-end protocol to guarantee service, or when it is acceptable for messages to be lost due to transmission errors.

Ordered Delivery Service

Messages can be sent specifying an optional ordered delivery queue. When ordered delivery is specified, messages with a given queue ID will be delivered to/from a given mobile device in the order in which they were sent. This option is